STORAGE VIRTUALIZATION OFFLOAD

BACKGROUND

[0001] Some types of storage devices have performance capabilities that can be difficult to fully utilize in some circumstances. Consider, for example, a computer having a CPU connected through a PCIe (Peripheral Component Interconnect Express) bus to an SSD (solid state device) that implements a version of the NVMe (Non-Volatile Memory express) logical device interface standard. The SSD's cost might have the same order of magnitude as the cost of a traditional disk drive, and yet in the same computer with the same high-speed bus, the SSD's latency and throughput performance might be an order of magnitude greater than a spinning type of disk drive. In other words, when attached through a high performance bus such as a PCIe bus, an SSD's latency and throughput can improve to the point where the storage device has fundamentally different characteristics than other types of block-based storage devices such as disk drives with spinning media.

[0002] The availability of high speed buses brings to the fore the performance differences between SSDs and traditional spinning disk drives. On a high speed bus such as a PCIe bus, an SDD's net latency and throughput can be significantly superior to that of a spinning disk drive. For example, an SSD attached through a PCIe bus might have a few microseconds of latency and might be capable of tens or hundreds of gigabits per second of throughput.

[0003] Much software for accessing storage devices has been designed with assumptions that persistent block-based storage will be relatively slow. For example, an operating system might be designed to deprioritize processes accessing storage, since they will likely have idle cycles while waiting for storage to respond. Also, because storage has been slow relative to processors and memory, complex memory-demanding caching schemes are often used to improve effective storage performance. Typically, the memory used for caching can add significant cost and power load to a computing system. If storage were able to be accessed at speeds close to processor speed, less memory and power would be required.

[0004] The lag of storage speed has affected the progress of virtualization technology. While some aspects of storage virtualization have been implemented in hardware, other aspects of storage virtualization discussed herein have lacked justification and have not previously been considered, since virtualizing in software has proven sufficient. Storage systems have not been able to provide sufficient data throughput to justify non-software virtualization solutions. In addition, merely throwing additional CPU cycles at an operating system or virtualization software will not necessarily improve performance. Devices such as NVMe SSDs can exchange data with a system at rates that can impact the system's CPU; CPU load generally increases with the rate of data exchange. As storage decreases in cost and therefore increases in amount, the high throughput rates of such devices will tax the host system. If a portion of a host's processing capacity is dedicated to handling storage, as storage increases, less processing becomes available for other purposes.

[0005] Moreover, some software is designed to limit storage latency or throughput. When a virtual machine, for example, requests access to storage, a delay might be built in because on average such requests are shortly followed by other requests. This deferment or batching of requests reduces the number of relatively slow switches between a hypervisor context and a virtual machine context. If a storage device and its attachment are capable of −30 us latency, an artificial 200 us batching delay reduces utilization of the storage hardware. And yet, if the virtualization software is tuned to work at 30 us, its CPU consumption could increase significantly (to handle the increased data throughput and access to the storage hardware).

[0006] It would be beneficial if there were convenient and cost-effective ways to improve storage virtualization efficiency. Techniques to that effect are described herein.

SUMMARY

[0007] The following summary is included only to introduce some concepts discussed in the Detailed Description below. This summary is not comprehensive and is not intended to delineate the scope of the claimed subject matter, which is set forth by the claims presented at the end.

[0008] Embodiments relate to off-loading aspects of storage virtualization to storage hardware and modifying software to take advantage of hardware virtualization features. A co-design of hardware and software allows a filesystem to provide files such that indirection overhead normally needed to access the content of files can be bypassed while still managing the files as filesystem objects. A storage device manages and exposes a virtual volume which is used to store the content of a file. Virtual volumes can be initialized or populated so that virtual blocks therein align with device storage blocks. A virtual volume can be initialized and populated by parsing a virtual disk file to access virtual disk metadata, which is then used to determine and set features of the virtual volume.

[0009] Many of the attendant features will be explained below with reference to the following detailed description considered in connection with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

[0010] The present description will be better understood from the following detailed description read in light of the accompanying drawings, wherein like reference numerals are used to designate like parts in the accompanying description.

[0011] FIG. **1** shows an example of a computing device with a storage software stack that provides virtualized block-based access to a virtual machine.

[0012] FIG. **2** shows details of how elements at different storage layers can perform indirection.

[0013] FIG. **3** shows an overview of how one or more layers of software indirection can be avoided without necessarily losing the conveniences of using a filesystem to manage and access a virtual disk file.

[0014] FIG. **4** shows a conceptual diagram of alignment between storage layers.

[0015] FIG. **5** shows a storage device with virtual volume features.

[0016] FIG. **6** shows an embodiment for configuring a virtual volume according to content in an existing filesystem.

[0017] FIG. **7** shows an embodiment for initializing a virtual volume by copying in data rather than allocating device blocks on the storage device.